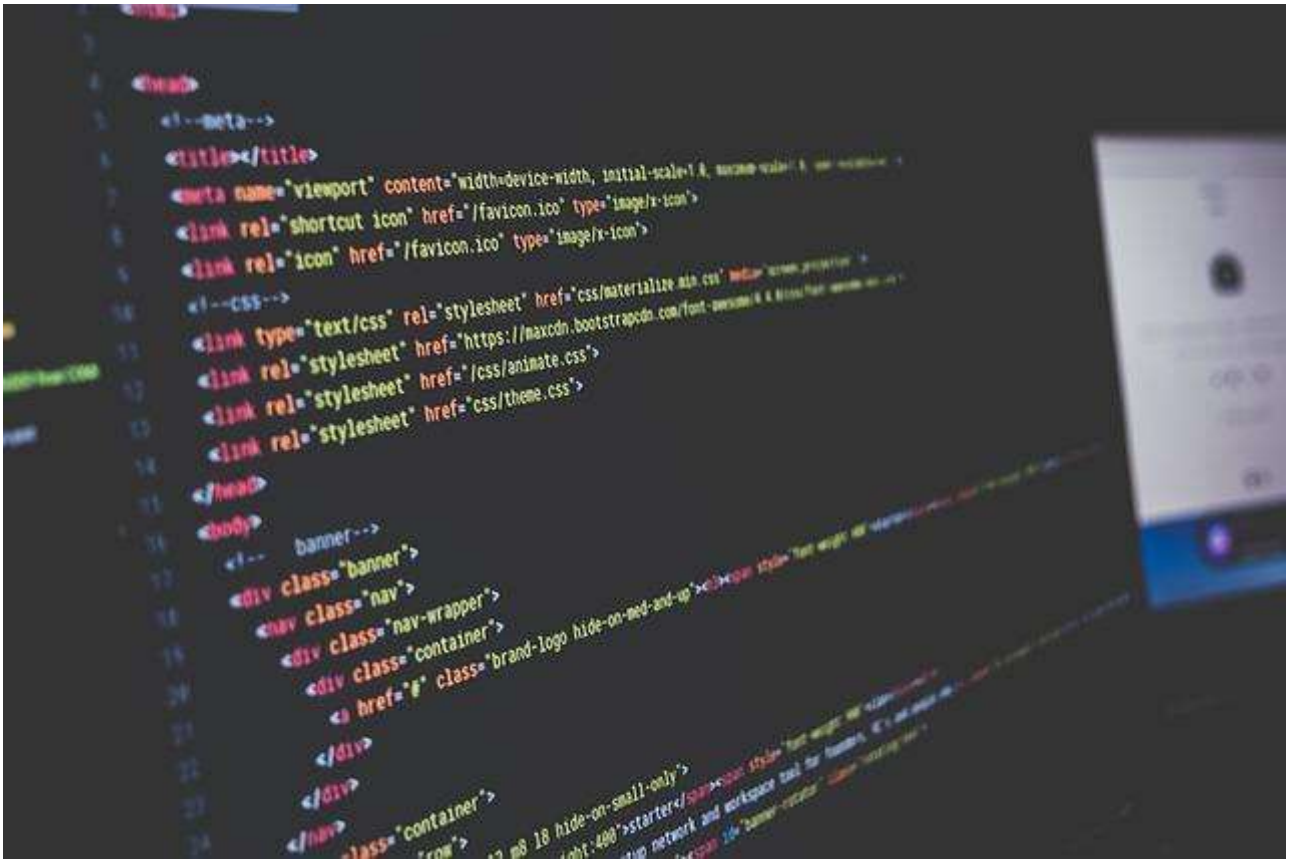
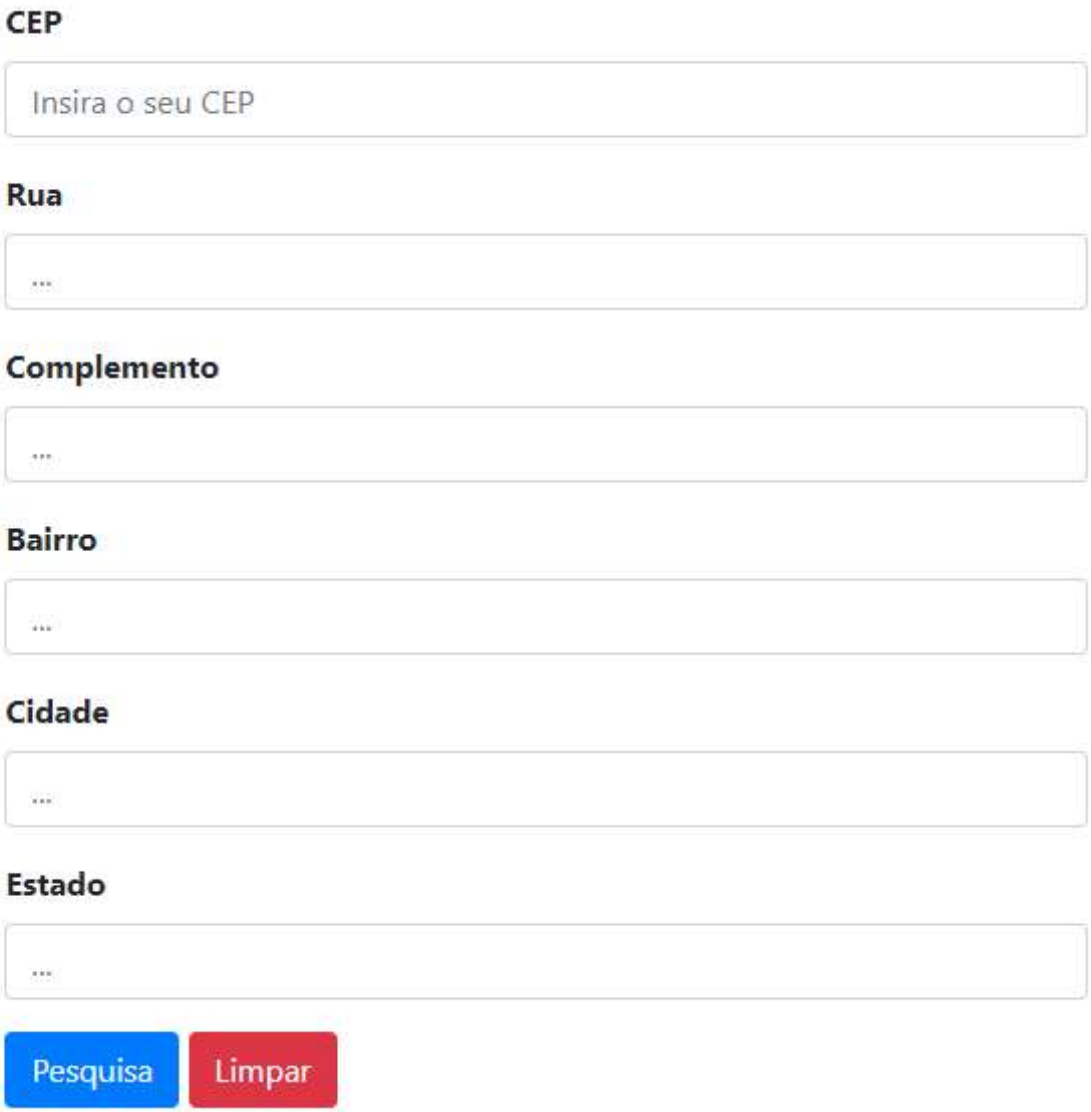
Empezando con fetch en Javascript

Elias Ribeiro
27/08/2021



Aprendas cómo utilicé el fetch para consumir datos de otros sistemas a través de Rest y completar automáticamente la siguiente lista:



*Subtitulo de la imagen.

CP

Dirección

Complemento

Barrio

Ciudad

Estado

Consulta Limpiar

Actualmente la persona que realizó la compra debe completar todos los campos correctamente para que haya la **validación** y la compra se finalice con éxito.

Sin embargo, es posible que algunos de los campos estén escritos incorrectamente, lo que resultará en una dirección incorrecta. Por lo tanto, tendremos problemas en la validación, finalización de la compra y la insatisfacción del cliente con la tienda porque no pudo realizar la compra.

Automatizando

Para automatizar el completado usaremos **Javascript**, veamos primer cómo **HTML** está configurado y tomar el atributo del button por Javascript.

```
<div class="container col-md-5">
     <form>
       <div class="form-group font-weight-bold">
         <label>CEP</label>
         <input type="text" id="cp" pattern= "\d{5}-?\d{3}" class="form-contr
       </div>
       <div class="form-group font-weight-bold">
         <label>Calle</label>
         <input type="text" class="form-control" id="calle" placeholder="..."
       </div>
       <div class="form-group font-weight-bold">
         <label>Complemento</label>
         <input type="text" class="form-control" id="complemento" placeholder
       </div>
       <div class="form-group font-weight-bold">
         <label>Barrio</label>
         <input type="text" class="form-control" id="barrio" placeholder="...
       </div>
       <div class="form-group font-weight-bold">
         <label>Ciudad</label>
         <input type="text" class="form-control" id="ciudad" placeholder="...
       </div>
       <div class="form-group font-weight-bold">
         <label>Estado</label>
```

```
          <input type="text" class="form-control" id="estado" placeholder="...
        </div>
        <button type="submit" id="btnConsultar" class="btn btn-primary">Consul
        <button type="button" id="btnLimpiar" class="btn btn-danger">Limpiar</
      </form>
  </div>
```

En Javascript se verá así:

```
const btnConsultarCEP = document.querySelector("#btnConsultar");
```

Una vez hecho esto, agregamos un evento de clic del botón usando **arrow function**:

```
btnConsultarCP.addEventListener("click", event =>{}
```

Implementemos algunas cosas dentro de esta función, primero **event.preventDefault**, responsable de cancelar el comportamiento natural del botón que sería una sumisión al servidor.

```
btnConsultarCP.addEventListener("click", event =>{    event.PreventDefault();
}
```

Con el comportamiento natural del botón cancelado, implementaremos la búsqueda del CP y tomaremos el valor que está contenido en el campo CP.

```
const inputDoCep = document.querySelector("#cp");

const valorDelCp = inputDelCp.value;
```

Una vez que hayas hecho esto, comencemos a trabajar con la **API** del Viacp y usamos el **fetch** por la URL para hacer una **solicitud**. Si se encuentra, haremos una función que devuelva la respuesta en **json**.

```
const inputDelCep = document.querySelector("#cp");

const valorDelCp = inputDoCep.value;const url = `https://viacp.com.br/ws/${val

fetch(url)
```

El `fetch` devuelve una promesa de que se retornará algo, esta promesa se llama **Promisse**. Esta promesa puede ser buena, haber devuelto los datos o haber fallado por alguna razón, como en el caso de caerse la **conexión con el servidor**.

Para ambas situaciones necesitamos hacer un tratamiento. En este caso, solo me ocuparé del éxito. Por eso, si es un éxito, entonces (**then**) tomaremos la respuesta con las informaciones del CP:

```
const inputDelCp = document.querySelector("#cp");
const valorDelCp = inputDelCp.value;
const url = `https://viacp.com.br/ws/${valorDelCp}/json/`;

fetch(url).then(response =>{
    return response.json();
        }).then(data =>
        {

        })
```

Con la devolución de la data (datos) lo asignaremos a los campos y haremos que una función tome el id de los campos y les asigne los valores:

```
function atribuirCampos(data)
{
const calle = document.querySelector("#calle");
const complemento = document.querySelector("#complemento");
const barrio = document.querySelector("#barrio");
const ciudad = document.querySelector("#ciudad");
const estado = document.querySelector("#estado");
```

```
calle.value = data.dirección;
complemento.value = data.complemento;
barrio.value = data.barrio;
ciudad.value = data.localidad;
estado.value = data.estado;
}
```

Y dentro del evento de clic del **btnConsultarCP** asignaremos la función:

```
const inputDelCp = document.querySelector("#cp");
const valorDelCp = inputDelCp.value;
const url = `https://viacp.com.br/ws/${valorDelCp}/json/`;

fetch(url).then(response =>{
return response.json();
    }).then(data =>{
        atribuirCampos(data);

       })
```

Pero como sabemos, el CP puede estar mal escrito, así que hagamos una excepción que muestre a la persona que era un CP incorrecto o inexistente.

Entonces creamos un `alert` dentro del then para mostrar que el CP no es válido.

```
.then(data =>
  {
  if(data.error)
 {
 alert("EL CP ESCRITO NO ES VÁLIDO");
 return ;
 } atribuirCampos(data);
})
```

Conclusión

En esta publicación, aprendemos cómo realizar una solicitud web con Javascript, viendo cómo estas solicitudes son útiles y cómo facilitan la vida a los desarrolladores. Y normalmente, en el día a día, los sistemas se comunican con otros por solicitudes.

Si te gustó el contenido, ¿qué tal si echas un vistazo en nuestra **formación front-end en Alura Latam**

Puedes leer también:

- HTML, CSS y Javascript, ¿cuáles son las diferencias?
- JavaScript: ¿Cuándo debo usar forEach y map?
- Cambiando CSS con JavaScript

ARTÍCULOS DE TECNOLOGÍA > FRONT END

- Estudia las 24 horas, los 7 días de la semana
- Foro y comunidad exclusiva para resolver tus dudas
- Acceso a todo el contenido de la plataforma por 6 meses

¡QUIERO EMPEZAR A ESTUDIAR!

Paga en moneda local en los siguientes países

ANUAL

US$79,90

un solo pago de US$79,90

- 218 cursos
- Videos y actividades 100% en Español
- Certificado de participación
- Estudia las 24 horas, los 7 días de la semana

Nuevos cursos
cada semana

NAVEGACIÓN

PLANES
INSTRUCTORES
BLOG
POLÍTICA DE PRIVACIDAD
TÉRMINOS DE USO
SOBRE NOSOTROS
PREGUNTAS FRECUENTES

¡CONTÁCTANOS!

¡QUIERO ENTRAR EN CONTACTO!

BLOG

PROGRAMACIÓN
FRONT END
DATA SCIENCE
INNOVACIÓN Y GESTIÓN
DEVOPS

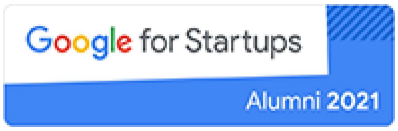
AOVS Sistemas de Informática S.A
CNPJ 05.555.382/0001-33

SÍGUENOS EN NUESTRAS REDES SOCIALES

ALIADOS

Empresa participante do SCALE UP ENDEAVOR

En Alura somos unas de las Scale-Ups seleccionadas por Endeavor, programa de aceleración de las empresas que más crecen en el país.



Fuimos unas de las 7 startups seleccionadas por Google For Startups en participar del programa Growth Academy en 2021

POWERED BY

CURSOS

Cursos de Programación
Lógica de Programación | Java

Cursos de Front End
HTML y CSS | JavaScript | React

Cursos de Data Science
Data Science | Machine Learning | Excel | Base de Datos | Data Visualization | Estadística

Cursos de DevOps
Docker | Linux

Cursos de Innovación y Gestión
Productividad y Calidad de Vida | Transformación Ágil | Marketing Analytics | Liderazgo y Gestión de Equipos | Startups y Emprendimiento